Chapter 9
IMAGE STEGANOGRAPHY USING DIFFUSION MODELS

Soham Singhal
Rohit Raval
Khushal Sharma
Vaishali Kulkarni

ABSTRACT
In our increasingly digital world, the demand for secure and discreet communication methods has never been greater. This project presents an innovative solution for secure message transmission by harnessing the power of image steganography along with generative adversarial networks and diffusion models. Our approach involves the forward diffusion of a concealed image into a cover image, allowing for the covert transmission of sensitive information between devices. When the concealed image is retrieved, it can be seamlessly reverse-diffused to reveal the hidden message. This method not only guarantees the confidentiality of the transmitted data but also preserves the visual integrity of the cover image, making it an effective and imperceptible means of secure communication in an era marked by heightened digital security concerns.

Keywords: *Sentiment Analysis; Naive Bayes; Support Vector Machine; Logistic Regression; Simple Neural Network; Convolutional Neural Networks; Long-Short Term Memory*

9.1 INTRODUCTION

Steganography, often referred to as the art of concealing a confidential message within an innocuous one, has long been a fascinating and valuable technique for covert communication[1]. In contemporary times, particularly in the realm of image steganography, there has been a notable evolution driven by the advent of generative adversarial networks (GANs). These advanced systems have revolutionized the process of seamlessly embedding hidden messages within cover images, offering an unprecedented level of discretion. Nevertheless, a critical security concern

looms over this approach—the absence of encryption for the concealed message itself. In response to this vulnerability, we introduce an innovative solution that capitalizes on recent developments in diffusion models. These models operate by incrementally introducing noise into input images through a meticulously designed diffusion process. Recent research has unveiled two indispensable characteristics of diffusion models that make them ideal for securing steganographic messages: the capability to effectuate translation between two images without the need for extensive training, and resilience in the face of noisy data. The models proposed in this chapter aim to bolster the security of steganographically concealed communications significantly. The key innovation lies in encrypting the secret message utilizing a diffusion model before integrating it into a cover image using GAN-based steganography techniques. By doing so, we not only harness the advantages of GANs for seamless embedding but also leverage the formidable cryptographic capabilities of diffusion models. This two-fold approach promises a substantial enhancement in safeguarding the secrecy of covert communications. The motivation behind exploring image steganography lies in the application of a robust method utilizing General Adversarial Networks (GANs) to conceal messages within cover images. The goal is to devise a unique model pipeline for seamlessly encoding a hidden image in an overlaying image. This technique holds potential for secure communication and data storage, finding applications in various fields. Embedding data in images offers protection against eavesdropping and theft. Moreover, image steganography's practical applications in numerous domains make it a versatile solution for ensuring data confidentiality and integrity. The aim is to enhance the security of image steganography by addressing vulnerabilities, such as the lack of encryption, through the use of diffusion models. Involving the conduct of in-depth research on image steganography techniques, the initiative encompasses exploring various methods, such as GAN-based techniques and diffusion models. This research endeavors to delve into existing algorithms, discern their strengths and weaknesses, and put forth improvements or innovative approaches. The overarching objective is to develop a model that incorporates the research findings, aiming to enhance image steganography approaches for secure communication and data retention

This chapter is organized as follows- Section 2 explores existing research on image steganography, diffusion models, and GANs, followed by details of the proposed model in section 3. Results are presented in section 4, followed by conclusion and references

9.2 LITERATURE REVIEW

Image steganography involves altering the cover image to conceal hidden data, ensuring it remains inconspicuous, unlike cryptography. Conversely, Steganalysis is em ployed to identify any concealed messages within the image and to retrieve the concealed information.[2] Steganalysis aids in

distinguishing between a stego image and a regular image. In addition to this classification, it also facilitates a deeper examination to pin- point the whereabouts and contents of the concealed image within the cover image. Due to the rapid progress of deep learning methods, numerous deep learning models have emerged for various tasks, including the field of image steganography. In this domain as well, there exists a diverse array of techniques and architectures tailored to accomplish this objective. Here we will be looking at some of the major approaches for the same. Image steganography employing CNN (Convolutional Neural Networks) models typically relies on the encoder-decoder structure. This framework takes two inputs, namely the cover image and the secret image, which are provided to the encoder to produce the stego image. Subsequently, this stego image serves as input to the decoder, which generates the secret image as output. While the fundamental concept remains consistent, different approaches have explored various architectural nuances. The specifics, such as the number of filters used, strides, filter sizes, activation functions, and loss functions, tend to differ from one method to another.

Image steganography leverages GANs (Generative Adversarial Networks) for their image generation capabilities. GANs consist of two primary components - a generator and a discriminator - engaged in a game-theoretic competition to create authentic-looking images. In the context of image steganography, GANs are applied to produce stego images through the concealment of secret data within cover images. The generator is responsible for generating these stego images, while the discriminator assesses the authenticity of the generated images. Some GAN-based techniques also incorporate a steganalyzer, which scrutinizes the stego images to detect any concealed information. Utilizing GANs in image steganography confers advantages such as enhanced security, resilience, and increased data-hiding capacity compared to traditional methods.

9.2.1 ***CNN based approaches***

Wu et al. [4] introduced an encoder-decoder structure. They employed a U-Net-based encoder-decoder architecture for concealment, and for extraction, they utilized a CNN consisting of six layers, as described by Duan et al. in [6]. These represent some of the initial architectural designs implemented to incorporate CNN networks. They used SSIM and MSE loss functions for their respective networks.

Further Baluja et al. [7] proposed a three network based approach. These networks were prep network, hiding network and reveal network. The prep network plays a role in preparing the secret image before it is input into the hiding network. The hiding network, in turn, takes the output from the prep network as well as the cover image to generate the container image. Subsequently, the reveal network deciphers the secret image from the container image, effectively revealing the cover image. The model computes two separate losses: one between the cover and

the constructed container, and another between the secret and the decoded image. Evaluation of the model's performance is carried out using the structure similarity index (SSIM). An extension to this approach, proposed by Zhang et al. in [8], introduces a modification called ISGAN. In this method, the cover image is transformed into a YCrCb image format, and the secret image is concealed exclusively within the Y channel, as all the semantic and color information is preserved in the Cr and Cb channels. This was essentially done to reduce the payload.

9.2.2 ***GAN based approaches***

Volkhonskiy et al. have introduced DCGAN[9] based Steganographic GAN (SGAN) which is a simple DCGAN with three modules - Generator, Discriminator and Steganalyzer. Three models engage in competition, where the generator creates stegoimages, the discriminator deciphers and retrieves the secret message, and the steganalyzer observes the probability produced by the generator.

One of most popular GAN models for image steganography SteganoGAN was proposed by Zang et al. [10]. SteganoGAN is a specialized Generative Adversarial Network (GAN) designed for image steganography, concealing secret data within cover images without human detection. It comprises three key components: a generator, a discriminator, and a steganalyzer. The generator embeds secret information into cover images to produce stego images, using the cover image and secret message as inputs. Conversely, the discriminator distinguishes genuine cover images from generated stego images, offering feedback to enhance stego image quality. An additional steganalyzer assesses stego image quality and seeks hidden information. This component enhances security and robustness in the steganographic process. SteganoGAN employs adversarial competition, with the generator striving for realistic stego images that mimic covers, while the discriminator attempts accurate classification.

Fu et al.[11] introduced an approach known as HIGAN, comprising three sub-networks: encoder, decoder, and discriminator. The encoder network, resembling a residual network in structure, is responsible for generating steganographic images with minimal color distortion and superior visual quality. The decoder network extracts secret images from these steganographic images. In contrast, the discriminator network enhances the steganography model's security through adversarial training in conjunction with the encoder-decoder network. The weights of the encoder-decoder network are optimized using mean square error (MSE) loss, while the weights of the discriminator network are optimized using bi- nary cross-entropy loss. Experimental findings demonstrate that the proposed HIGAN model effectively produces high-quality steganographic images and enhances the security of the steganography technique.

9.2.3 *Diffusion based approaches*

Yu et al.[12] introduced a novel approach, to enhance the security of steganography through the utilization of diffusion models called CRoSS. By harnessing the strength of diffusion models in the domain of image translation, they employed textual prompts to transform the hidden image into a different form. In the hiding process, they employed a conditional diffusion model to convert a secret image into a container image, employing both forward and backward processes. In the revealing process, the container image underwent processing using the diffusion model's backward process to extract the revealed image. This method involves the generation of two keys: a private key, which characterizes the content within the secret image, and a public key, which governs the content within the container image.

9.3 PRELIMINARIES

In this section, we lay the groundwork for the technical aspects of our initiative. This includes, an explanation of core concepts such as GANs and diffusion models, and a thorough analysis of our dataset.

9.3.1 *GAN model*

Generative Adversarial Networks or GANs utilize machine learning to autonomously find and learn the underlying structures and forms in input information. They do this in a manner that allows the model to generate or produce new instances that realistically could have originated from the initial dataset, even though they may not have actually been present within it. GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to clas sify examples as either real (from the domain) or fake (generated)[3]. In our approach, we employ Generative Adversarial Networks (GANs) to perform image steganography.

9.3.2 *Diffusion Models*

A diffusion model is a probabilistic generative model used in various fields, including ma- chine learning, image processing, and data generation. This model is designed to capture and model complex data distributions by iteratively diffusing or spreading information across data points in a sequential manner. What we particularly want to implement, is a specific set of models called Denoising Diffusion Probabilistic Models (DDPMs).

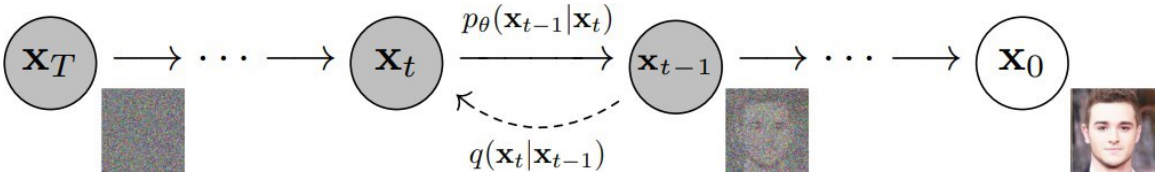

Figure 9-1 Forward and Reverse Diffusion [4]

Fig 9-1 shows the forward and reverse diffusion. The model is made up of two phases:

- Forward diffusion q that gradually adds random noise in each iteration.
- Reverse diffusion pè, where a neural network works to remove the noise to return back to the original image.

9.3.3 ***Dataset Analysis***

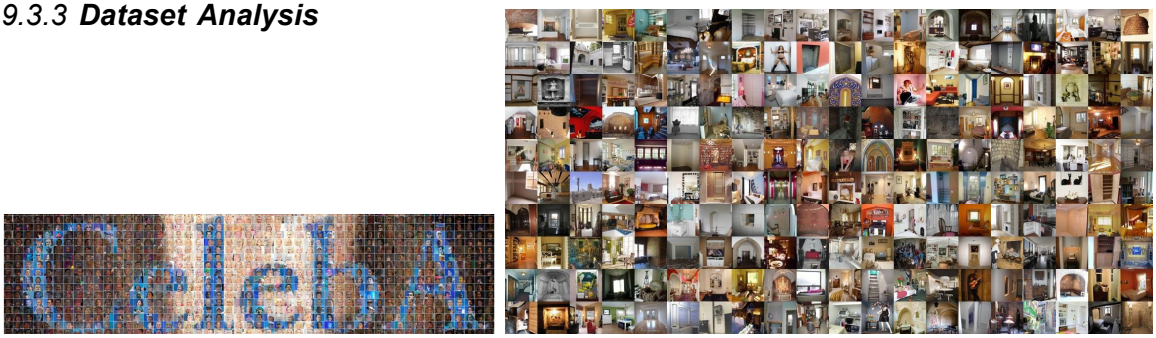

Figure 9-2 Datasets used (a) CelebA Dataset and (b) Places Dataset

In steganography, we use the Places dataset for cover images and the CelebA dataset for concealed messages. The Places dataset, with over 2 million diverse, high-resolution images, provides a visually authentic canvas for discreetly hiding information. Spanning various scenes globally, it's ideal for concealing data while preserving the natural appearance of the cover image.

In contrast, the CelebA dataset, with over 200,000 celebrity face images, serves as the message to be diffused. Its collection of facial identities and expressions makes it valuable for embedding secret messages within subtle features. This approach has potential applications in privacy-preserving communication and watermarking. By combining the strengths of these datasets, we achieve effective and visually imperceptible steganography

9.4 METHODOLOGY

The previous section explored the fundamentals of steganography and diffusion model, and also discussed the dataset intended for training the model. Here the focus is on the methodology and

architecture. The flowchart of our approach is illustrated in the figure below In simple terms, our plan involves employing a diffusion model to encrypt an image by introducing noise to it.

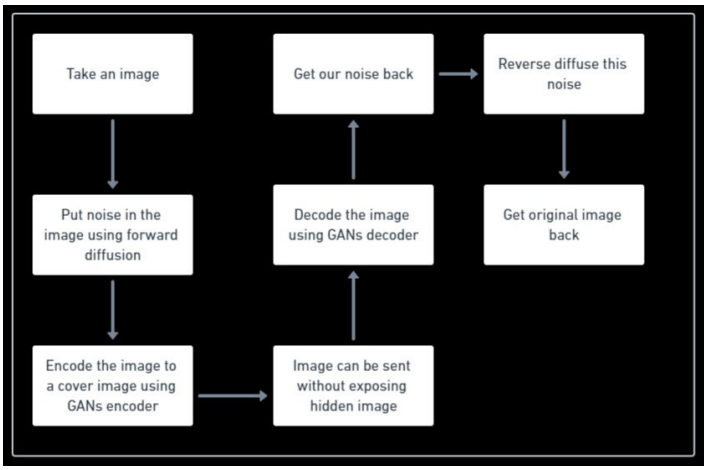

Figure 9-3 Illustration of proposed approach

This noise will be governed by a Gaussian algorithm as detailed below. The resulting noisy image will serve as input to our GANs network, which will conceal it within a cover image.

On the receiver's end, the concealed noisy image will be extracted, and through the reverse diffusion process, we will identify and recover the noise. This approach combines the utilization of diffusion and GANs to establish a robust method of communication.

9.4.1 Diffusion Models

As mentioned before, the diffusion process is subdivided into two parts - Forward diffusion which adds noise and Reverse diffusion which removes noise.

9.4.1.1 Forward Diffusion

The forward diffusion process gradually adds noise to an image from the real distribution, in a number of time steps T. This happens according to a variance schedule.[5]. When expressed mathematically, we have the following equation:

$$q(x_t|x_{t-1}) = N(x_t; \sqrt{(1-\beta t)}x_{t-1}, \beta_t I) \qquad (4.1)$$

This equation delineates the probability distribution governing the evolution of the variable xt over time within a forward diffusion model. As evident, the current state of the variable is influenced by its previous state, which is represented by the conditional probability q(xt|xt-1).

The variable ât plays a pivotal role in determining the degree of noise added at each time step. It can be determined through various methods, such as a quadratic equation or a cosine equation. In our implementation, a linear equation is employed.

Comparing this equation to the standard normal distribution equation, we can deduce that the mean μt is given by $\sqrt{(1-\beta t)}xt-1$, and the variance σ2 is equal to βt.

Since the sum of Gaussian, is just another Gaussian, equation 4.1 can be written as:

$$q(x_t \mid x_0) = N(x_t; x_0, (1-\bar{\alpha}_t)I) \qquad (4.2)$$

where αt := 1 βt and a¯t := $\prod_{s=1}^{t} \alpha_s$

Now that we've established the maths behind forward diffusion, let us look at a pseudo code that we implemented.

Algorithm 4.1 forward diffusion

Require: x_0 (tensor), t (int)

1: Sample random noise with shape of x_0

2: Get $\sqrt{alphas\_cumprod}$ at time step *t.*

3: Get $\sqrt{1-alphas\_cumprod}$ at time step *t.*

4: Move tensors to device.

5: Apply forward update.

6 : **return** $x_1 = x_t, noise\ x_t = \sqrt{alphas\_cumprod}.x_0 + \sqrt{1-alphas\_cumprod}.noise$

9.4.1.2 ***Reverse Diffusion***

Unlike forward diffusion, reverse diffusion is a bit more complicated and involves the use of an autoencoder neural network. The mathematical equation governing the reverse diffusion is as follows:

$$p_e(x_{t-1} \mid x_t) = N(x_{t-1}; \mu_e(x_t, t), \Sigma_e(x_t, t)) \qquad (4.3)$$

Here the mean and variance are known at time step t. Our neural network needs to learn/represent the mean and variance. The neural network needs to take in a noised image at a particular time step and return the predicted noise[5]. A UNET model architecture can be used for this application. The following figure shows what a UNET architecture looks like.

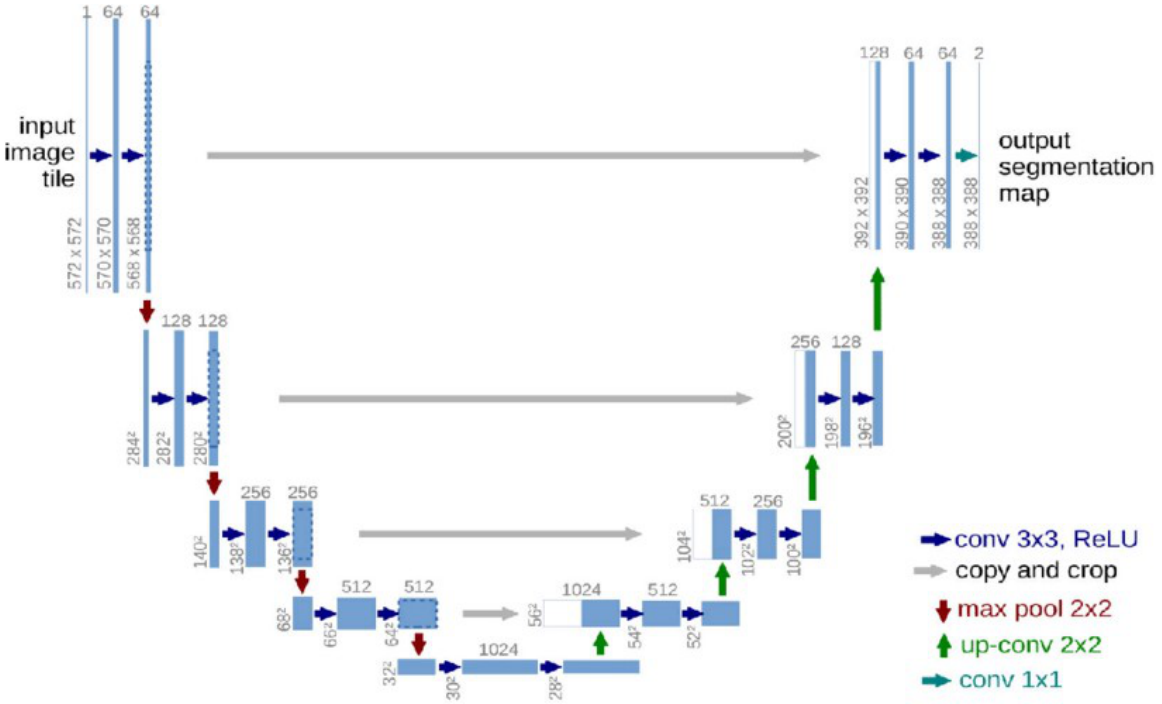

Figure 9-4 Example of UNET architechture [5]

As can be seen from Fig 9-4, a U-Net model first downsamples the input (i.e. makes the input smaller in terms of spatial resolution), after which upsampling is performed. As param eters of the neural network are shared over time, we need to send in the timestamp as an input along with the noisy image. For this, we make use of a concept from positional embeddings. We encode timestep *t* into positional embeddings of dimension *dim* such that the *t* can be represented by a tensor of size *(batch_size, dim).*

9.4.2 GAN Network

GAN models typically consist of two main components: a generator, responsible for creating synthetic data samples that closely resemble real data, and a discriminator, which evaluates the input data to differentiate between real and fake samples. However, for our specific use case, we employ three distinct components: a discriminator, an encoder, and a decoder, as shown Fig 9-5.

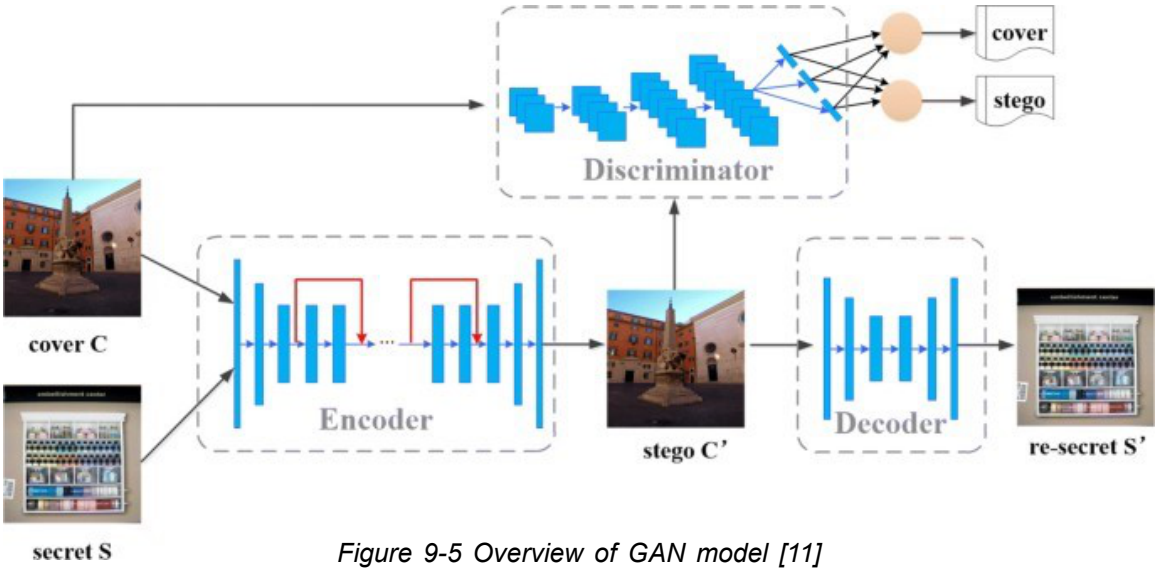

Figure 9-5 Overview of GAN model [11]

9.4.3 ***Encoder***

The encoder consists of down-sampling layers, residuals layers and up-sampling layers[11]. The structure of the encoder model is given in Fig 9-6. The network utilizes skip connections that blend the shallow and the deep features across convolution stages. These features can be useful to generate steganographic images.

GANs are notorious for being extremely unstable during the adversarial training process and tend to have disappearing gradients. To tackle this problem we add residual blocks to our network. Each convolution is followed by a batch normalisation layer and ReLu activation. Within the intermediate residual layers, there are nine consecutive residual blocks aimed at acquiring a broader range of both low-dimensional and high-dimensional features. Each individual residual block consists of two 3×3 convolutional layers, each with a stride of 1, accompanied by a skip connection. Additionally, a dropout layer is introduced between these two convolutional layers.

9.4.4 ***Decoder***

The decoder network composed of a 6-layer full convolutional network extracts the secret colour images (S') from steganographic images (C')[11]. Previous research has demonstrated the efficacy of the decoder network's design in successfully reconstructing both single-channel grayscale and three-channel color secret images. The network comprises 3×3 convolutional layers with a stride of 1 and padding of 1, each followed by batch normalization (BN) and the ReLU activation function. However, it's worth noting that a sigmoid activation function was applied after the final convolutional layers.

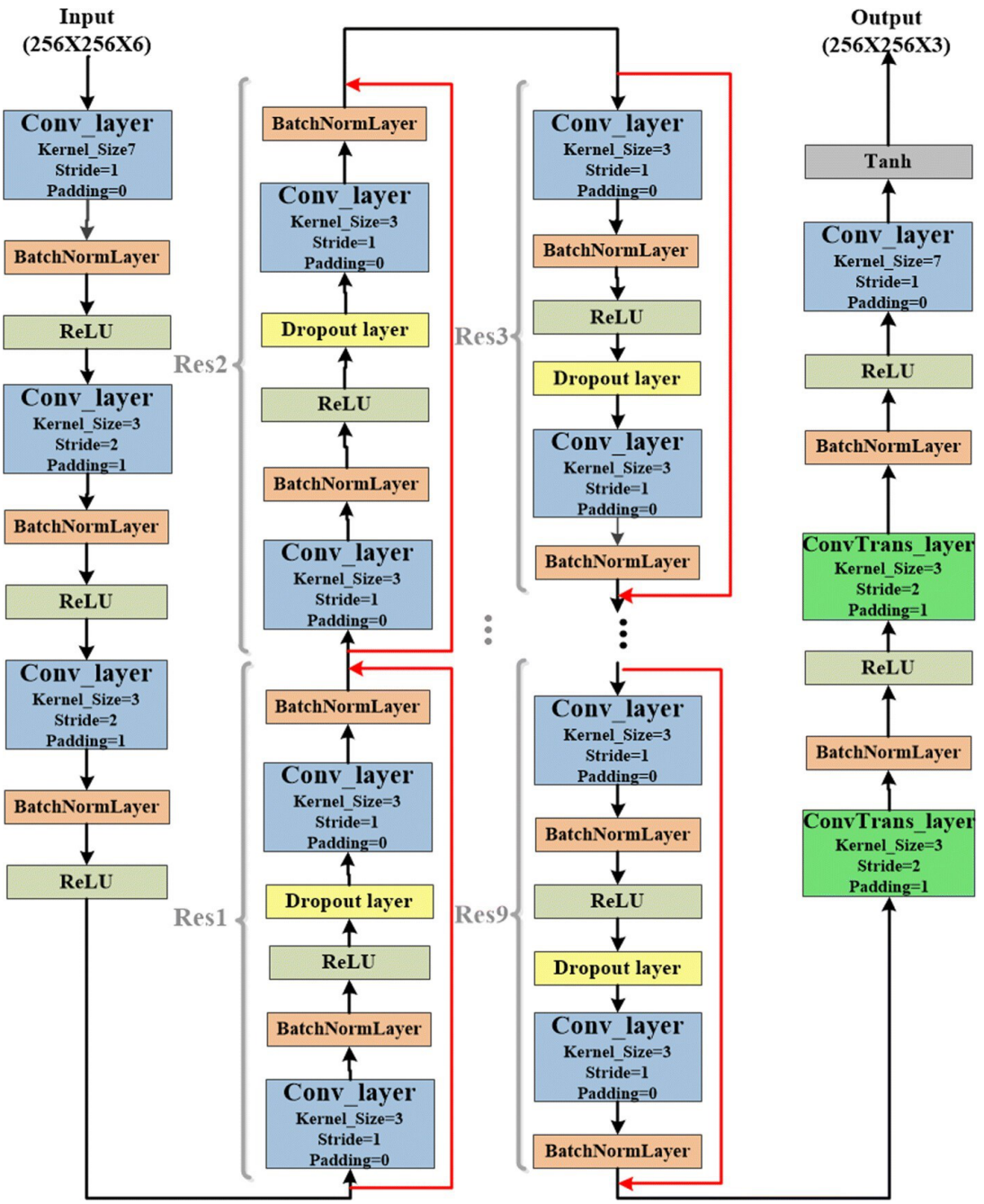

Figure 9 6 Architecture for Encoder [11]

9.4.5 *Discriminator*

Previous methods of steganography based on deep learning used to hide the color three- channel secret image just focused on improving the visual quality of steganographic images and revealed secret images. The literature [8], suggested an enhanced steganography model that concealed the grayscale secret image within the Y channel of the coloured carrier image. The model can be used as a binary discriminator to determine whether the cover images contain secret information or not. For our initiative, we have used an advanced discriminator model called Xu'Net. Xu'Net is a popular image-based discriminator used frequently in image steganalysis. The XuNet architecture was proposed by Xu et al. and has been modified and improved upon by other researchers. The focus of XuNet steganalysis is on the filter and network architecture, and it has been shown to outperform other methods in detecting steganography in digital images.

9.5 IMPLEMENTATION AND TOOLS USED

We have implemented the diffusion models and GAN architectures using the PyTorch library. Our implementation encompasses both the encoding and decoding processes, crucial components of our innovative steganography approach. Given below are the visual representations of the models we created along with explanations

9.5.1 *Forward Diffusion*

Our first major milestone involved the implementation of the forward diffusion process, which is fundamental to our steganography methodology. These models are responsible for converting hidden images into noise, a vital initial step in introducing robust security to the data hiding process.

The forward diffusion process can be directly replicated without the need for a deep learning model by applying mathematical equations, as previously detailed. We utilized this algorithm and put it into practice to generate a fully noisy image.

9.5.2 *GAN based encoder-decoder model*

Parallelly, we have integrated GANs into our steganography pipeline. Specifically, we have utilized a GAN-based encoder-decoder framework for the encoding and decoding processes. This integration allows us to seamlessly embed noise patterns (representing hidden data) into cover images while maintaining a natural appearance, a critical aspect of our steganography technique. Figure 5.2 depicts the structures of various components within the GAN model.

9.5.3 Encoder

The primary objective of the encoder segment is to generate a stego image that conceals hidden data within the cover image while visually resembling the cover image. This is achieved by combining a hidden (noisy) image with the cover image. Mean Square Error (MSE) is used as the loss function during training, quantifying the pixel-wise difference between the stego image and the true cover image. The training process aims to minimize this loss, ensuring that the stego image closely matches the cover image in terms of pixel values. This can be represented with the following equation:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(I_{stego}(i) - I_{cover}(i))^2 \qquad (5.1)$$

MSE : Represents Mean Square Error, the loss function.
N : Denotes the total number of pixels in the images.
$I_{\text{stego}}(i)$: Represents the pixel value of the stego image at pixel *i.*
$I_{\text{cover}}(i)$: Represents the corresponding pixel value of the cover image at pixel *i.*

9.5.4 Decoder

As discussed in the preceding chapter, the primary objective of the decoder is to extract the obscured hidden message from the concealed image. Similar to the encoder, it too employs the Mean Square Error (MSE) as a loss function to steer the training process. Within this model, the MSE loss assesses the mean squared disparity between the pixel values of the retrieved secret images (S') and the original secret images (S). This quantification gauges the decoder's proficiency in restoring the initial secret information.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(S'(i) - S(i))^2 \qquad (5.2)$$

MSE : Represents Mean Square Error, the loss function.
N : Denotes the total number of pixels in the images.
$S^2(i)$: Represents the pixel value of the extracted secret image at pixel *i.*
$S(i)$: Represents the corresponding pixel value of the original secret image at pixel *i.*

9.5.5 Discriminator

The discriminator in a Generative Adversarial Network (GAN) is trying to distinguish between real data (in this case, real stego images) and fake data (generated stego images produced by the generator). It uses Binary Cross-Entropy (BCE) as the loss function. Specifically, the goal of this loss calculation is to encourage the generator to produce stego images that are convincing enough to fool the discriminator into classifying them as real.

$$\mathrm{BCE}(y, p) = -y\log(p) + (1-y)\log(1-p) \qquad (5.3)$$

Where:

y : Represents the true label (1 for real, 0 for fake).

p : Represents the predicted probability assigned by the discriminator.

9.5.6 *Training of GANs*

The complete code for training the GAN model has also been implemented. During training, the encoder and decoder are updated in every epoch, whereas the discriminator undergoes updates every 2 epochs. The weighted losses of all models are combined, and gradients are subsequently calculated. To calculate gradients, the decoder loss is multiplied by a factor called beta, and the discriminator loss is multiplied by another factor called gamma. These are hyperparameters, and we've assigned values of 0.75 for beta and 1 for gamma in this case. The encoder and decoder are trained using the Adam optimizer with a learning rate of 0.001, while the discriminator is trained using the SGD optimizer with a learning rate of 0.0005.

9.5.7 *Reverse Diffusion*

In our study, we have explored diverse architectural configurations of UNet for the purpose of reverse diffusion. Specifically, our choice of employing the fundamental, preliminary UNet structures was primarily motivated by considerations related to resource constraints and the inherent complexity associated with alternative architectures. At each step, the model requires a time-step embedding, which is provided as input for every forward step of the UNet model.

For training, we use the L1 loss (Mean Absolute Error) to measure the loss between the actual noise and the predicted noise. By minimizing this loss, the model is trained to effectively reverse the diffusion process and reconstruct the original data. This is the primary objective of the reverse diffusion model.

$$L1 = \frac{1}{N}\sum_{i=1}^{N}\left|noise(i) - noise\_pred(i)\right| \qquad (5.4)$$

Where:

N : Denotes the total number of elements in the noise. Noise

(i) : Represents the actual noise at element i.

noise pred(i) : Represents the predicted noise at element i.

9.5.8 *Tools Used*

During this endeavor, a range of tools were employed to facilitate its execution. Notably, two

primary tools played a pivotal role in our workflow: Jupyter Lab for coding and Jarvis AI for the A-100 GPU.

9.5.8.1 Jupyter Lab

Jupyter Lab is a versatile online integrated development environment (IDE) that's particularly well-suited for running Python code. In Jupyter Lab, you work with code in "cells", allowing you to write and execute Python code step by step. As an online IDE, it's accessible directly from your web browser, eliminating the need for local installation.

9.5.8.2 Jarvis Lab

In the initial stages, we used the T4 GPU from Google Colab to train our models. While the forward diffusion phase was successful, challenges arose during GAN and reverse diffusion training due to convergence issues and limited RAM. To address these, we switched to the A-100 GPU with significant computing power, resulting in reduced model convergence time. Hosted online, the A-100 GPU extended our available RAM, contributing to a more efficient training process. Table 9.5.1 shows the comparison between NVIDIA A-100 and T4 GPUs.

Table 9-1 Comparison between NVIDIA A-100 and T4 GPUs

Feature	NVIDIA A-100	NVIDIA T4
CUDA Cores	Up to 6,912	2,560
Tensor Cores	Yes	No
Memory	40 GB or 80 GB HBM2	16 GB GDDR6
Memory Bandwidth	Up to 1.6 TB/s	Approx. 320 GB/s
Tensor Core FLOPs	Up to 312 TFLOPs	N/A
Performance	High-performance, suitable for HPC, AI	Suitable for mainstream AI and ML tasks
Price	Relatively expensive	More affordable

9.6 RESULTS AND ANALYSIS

The results of our work can basically be divided into three segments - the forward diffused image result, the GANs networks result and the Reverse diffusion result.

9.6.1 Forward Diffusion

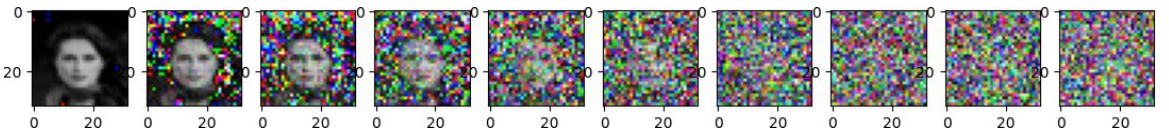

Figure 9-7 Forward diffusion with timestep 100

Fig 9-7 shows the forward diffusion result with a timestamp of 100. In this approach, choosing an appropriate beta value is crucial for the overall image transformation. A broad range of beta values provides flexibility to tailor the diffusion process to specific requirements. Smaller beta values control noise accumulation, resulting in a gentle evolution of the image. Larger beta values expedite noise incorporation, leading to a rapid transformation. This adaptability allows fine-tuning the steganographic process to balance security and visual fidelity based on the application's needs.

To manage computational resources effectively, a practical solution involves gradually introducing noise to a 128x128 image, followed by upscaling to 256x256 dimensions. This approach addresses potential RAM limitations, ensuring the diffusion process aligns with the GAN's input requirements. It demonstrates an efficient strategy to maintain data integrity while accommodating computational constraints.

*9.6.2 **GANs Model***

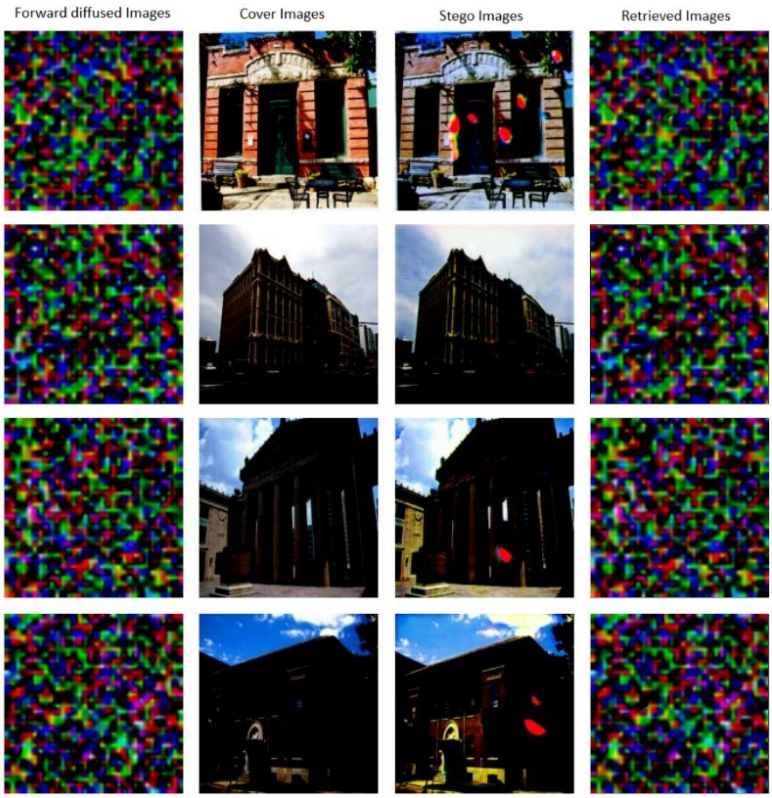

Figure 9-8 From left to right- Forward diffused image, cover image, stego image, extracted image

After undergoing training for a hundred epochs, each epoch taking around 1 min 30 sec, our GANs model produces the following outcomes. Visually, it is evident that the encoding process is executed with a high degree of precision, making it exceedingly challenging to differentiate between the stego image and the cover image. This is backed by the low encoding loss of 0.005. The extraction by the decoder also has high degree of precision and it is backed by its low 0.10 loss. However, the discriminator exhibits relatively higher loss, specifically at 0.7.

To address this, one potential solution involves an extended training duration and a modification in the frequency of discriminator weight updates. These adjustments may lead to enhanced accuracy and a reduction in loss, ultimately working towards the optimization of the model's performance. Fig 9-8 shows the result from from left to right - Forward diffused image; Cover Image; Stego Image; Extracted image.

9.6.3 ***Reverse Diffusion Model***

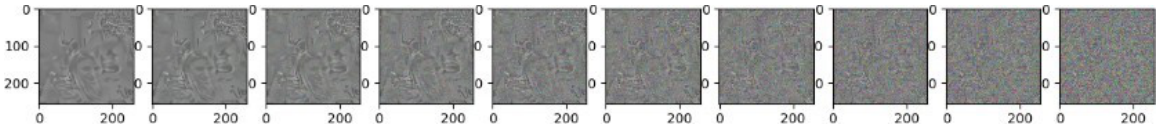

Figure 9-9 Reverse diffusion trained 200 epochs

The reverse diffusion model is meticulously trained over 200 epochs, each taking just over 1 minute and 30 seconds on a high-performance GPU. Fig 9-9 shows the result of reverse diffusion. The primary goal is to reverse the diffusion process and denoise the forward-diffused image. However, around the 100-epoch mark, it becomes apparent that the model's attempts do not achieve the desired accuracy, evident from the loss function oscillating between 0.2 and 0.1. Factors contributing to this include large datasets challenging convergence time and limited hardware resources.

The model may benefit from an extended training duration for improved accuracy and stability, beyond current constraints. The model's architecture, a basic UNET model prioritizing objectives other than the highest accuracy, affects performance. Addressing limitations involves exploring alternative architectures, additional training time, and optimizing hardware resources for more successful reverse diffusion and denoising.

9.6.4 ***Evaluation Matrices***

Evaluation matrices typically refer to tools and methods used to assess the performance of machine learning models. These matrices help quantify how well a model is performing in

various tasks. The two evaluation matrices that we utilize are Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

9.6.5 ***Peak Signal-to-Noise Ratio (PSNR)***

Peak Signal-to-Noise Ratio (PSNR) is a widely used image and video quality metric, gauging the degradation in quality of a reconstructed or compressed image or video compared to its original version. It quantifies noise and distortion, with higher PSNR values indicating better quality. Our model yielded an average PSNR value, suggesting reasonable quality with some distortion compared to the originals. Higher PSNR values imply better image quality, leaving room for improvement.

9.6.6 ***Structural Similarity Index (SSIM)***

The Structural Similarity Index (SSIM) is a commonly used metric in image and video processing to assess image quality. It measures the similarity between a reference image and a distorted or compressed one, considering factors like luminance, contrast, and structure. SSIM provides a more comprehensive evaluation compared to pixel-wise comparisons. Our models calculated mean SSIM, indicating significant dissimilarity between the original and compared images in terms of structure and content. This suggests that the compared images may not closely resemble the originals.

Table 9-2 PSNR and SSIM of GANs calculated after 100 epochs

Metric	Value
PSNR (Peak Signal-to-Noise Ratio)	27.89
SSIM (Structural Similarity Index)	0.05

In image processing, higher PSNR and SSIM values (closer to 1) are generally desirable. Acceptable values vary by application and image nature. To achieve higher quality, explore and fine-tune processing or compression techniques. Visually inspect compared images to understand distortion and determine if it aligns with analysis requirements. For improvement, optimize the processing pipeline and adjust parameters to enhance image quality and similarity to the originals.

9.7 CONCLUSION AND FURURE SCOPE

9.7.1 ***Conclusion***

Our fusion of diffusion models and Generative Adversarial Networks (GANs) in advancing steganography reveals an innovative landscape with challenges. Findings highlight the promise of securely concealing images within others, supported by visually convincing stego images and low encoding loss, affirming precise encoding for secure data transfer. Resource constraints,

including GPU and RAM limitations, led to delays and necessitated management strategies. Downsampling and upscaling to address RAM issues introduced risks of data loss and image quality degradation. Adapting to knowledge gaps revealed potential inaccuracies in convolution and deconvolution layers. In conclusion, our work underscores the innovative potential in steganography, emphasizing resource management, adaptability, and precision. These insights serve as guiding beacons for robust, efficient, and secure steganographic techniques, marking a step towards a new era in secure data transfer through the fusion of diffusion models and GANs.

9.7.2 Future Scope

The future of our steganography approach is brimming with possibilities, presenting numerous avenues for refinement and expansion. As we chart a course forward, we envision several key areas of development that hold the potential to enhance the accuracy and robustness of our proposed approach.

Enhancing Accuracy and Robustness: Strengthening our steganography technique involves exploring strategies to fine-tune and optimize the model. This includes extending training duration and adjusting discriminator weight update frequency for improved accuracy and reduced loss, ensuring the extracted image closely resembles the original hidden image.

Consistency Models: Drawing inspiration from recent research, like OpenAI's "Consistency Models," shows potential for making diffusion model outputs consistent and more robust against added noise. Integrating these models into our approach could substantially enhance resilience and accuracy, reinforcing the security and reliability of our steganographic technique.

Key Component Enhancement: Another avenue for development is adding a key component to the diffusion model. This component can make the model consistent with a unique key, facilitating accurate decoding of the originally distorted hidden image. This approach offers exciting possibilities for creating a secure and adaptable steganography technique, allowing for greater flexibility and customization.

Harnessing the full potential of our innovative fusion of diffusion models and GANs requires ongoing research and development. This groundbreaking approach to steganography, if utilized effectively, can advance data concealment security and extend to fields prioritizing accurate and secure visual information transmission. The journey ahead promises more robust and efficient steganography, aligning with the evolving landscape of information security and privacy.

Embarking on this future scope, we are guided by the belief that continuous innovation and

adaptation are the cornerstones of advancing steganography in a dynamic digital landscape. Our unwavering commitment to staying at the forefront of this field drives us to explore new horizons and unlock the full potential of our fusion approach.

REFERENCES

[1] Liu J., Ke Y., Zhang Z., Lei Y., Li J., Zhang M., Yang X. (2020). "Recent Advances of Image Steganography With Generative Adversarial Networks", in IEEE Access, vol. 8, pp. 60575-60597, doi: 10.1109/ACCESS.2020.2983175.

[2] Shi H., Zhang X., Wang S., Fu G., Tang J. (2019). "Synchronized detection and recov- ery of steganographic messages with adversarial learning", in International Conference on Computational Science, pages 31–43. Springer

[3] Brownlee, J. (2019). "A gentle introduction to generative adversarial networks (Gans)". MachineLearningMastery.com

[4] Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. arXiv preprint arXiv:2006.11239.

[5] Rogge, N., Rasul, K. (2022). "The Annotated Diffusion Model". Hugging Face– The AI community building the future. https://huggingface.co/blog/annotated- diffusion#the-annotated-diffusion-model

[6] Wu P., Yang Y., Li X. (2018). "Image-into-image steganography using deep convolu- tional network", in Pacific Rim Conference on Multimedia, pages 792–802. Springer

[7] Baluja S. (2017). "Hiding images in plain sight: Deep steganography", in Advances in Neural Information Processing Systems, pages 2069–2079

[8] Zhang, R., Dong, S., Liu, J. (2019). "Invisible steganography via generative adversarial networks". Multimed Tools Appl 78, 8559–8575. https://doi.org/10.1007/s11042-018- 6951z

[9] Volkhonskiy D., Nazarov I., Burnaev E. (2019). "Steganographic generative adversar- ial networks" arXiv:1703.05502

[10] Zhang K., Cuesta-Infante A., Xu L., Veeramachaneni K. (2019). "SteganoGAN: High Capacity Image Steganography with GANs", , arXiv:1901.03892

[11] Fu Z., Wang Y., Cheng X. (2020). "The secure steganography for hiding images via GAN". J Image Video Proc. 2020, 46. https://doi.org/10.1186/s13640-2020-00534-2

[12] Yu J., Zhang X., Xu Y., Zhang J. (2023). "CROSS: Diffusion Model makes control- lable, robust and secure image steganography".arXiv:2305.16936

[13] Li B., Xue K., Liu B., Lai Y. (2023). "BBDM: Image-to-Image Translation with Brownian Bridge Diffusion Models".arXiv:2205.07680.

[14] Song Y., Dhariwal P., Chen M., Sutskever I. (2023). "Consistency Mod- els".arXiv: 2303.01469.